A Sketch: Solving a Multidimensional Three Phase Flow Problem

Hans Fredrik Nordhaug
Ann Kristin Ask
Helge K. Dahle
Kenneth Hvistendahl Karlsen

March 10, 2000



Outline

- Short introduction
- Operator splitting
- Advection solver
  - Streamlines
  - Front tracking
- Conclusion

Introduction

We formulate the three phase flow problem as the following 2×2 system of parabolic equations:

$$\begin{aligned} \partial_t u + \nabla \cdot (f(u)\mathbf{q}) &= \varepsilon \nabla \cdot (D(U)\nabla \cdot u), \\ \partial_t v + \nabla \cdot (g(u,v)\mathbf{q}) &= \varepsilon \nabla \cdot (D(U)\nabla \cdot v), \\ U(\mathbf{x},0) &= U_0(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n,\ t > 0 \end{aligned} \tag{1}$$

where $U = [uv]$. The velocity field $\mathbf{q}$ is assumed to satisfy $\nabla \cdot \mathbf{q} = 0$.

We will solve this system using operator splitting, i.e., sequentially solving the advection part and the diffusion part.

Operator splitting

We will solve Equation (1)) using operator splitting as follows:

Step 1: Solve the advection part up to time $t = \Delta t$.

$$\begin{aligned} \partial_t u + \nabla \cdot (f(u)\mathbf{q}) &= 0, \\ \partial_t v + \nabla \cdot (g(u,v)\mathbf{q}) &= 0, \\ U(\mathbf{x}, 0) &= U_0(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^n,\ t > 0 \end{aligned} \tag{2}$$

Step 2: Solve the diffusion part up to time $t = \Delta t$ with the solution from the advection step as initial data.

$$\begin{aligned} \partial_t u^{\Delta t} &= \varepsilon \nabla \cdot (D(U^{\Delta t}) \nabla \cdot u^{\Delta t}), \\ \partial_t v^{\Delta t} &= \varepsilon \nabla \cdot (D(U^{\Delta t}) \nabla \cdot v^{\Delta t}), \\ U^{\Delta t}(\mathbf{x}, 0) &= U(\mathbf{x}, \Delta t), \quad \mathbf{x} \in \mathbb{R}^n,\ t > 0 \end{aligned} \tag{3}$$

Step 2: Solve the diffusion part up to time $t = \Delta t$ with the solution from the advection step as initial data.

$$\begin{aligned} \partial_t u^{\Delta t} &= \varepsilon \nabla \cdot (D(U^{\Delta t}) \nabla \cdot u^{\Delta t}), \\ \partial_t v^{\Delta t} &= \varepsilon \nabla \cdot (D(U^{\Delta t}) \nabla \cdot v^{\Delta t}), \\ U^{\Delta t}(\mathbf{x}, 0) &= U(\mathbf{x}, \Delta t), \quad \mathbf{x} \in \mathbb{R}^n,\ t > 0 \end{aligned} \tag{3}$$

Step 3: Use the solution from the diffusion step as initial data for the advection step, i.e, set $U_0 = U^{\Delta t}$. Continue this loop until final time is reached.

Advection solver

We will solve Equation (2) along streamlines, $\mathbf{r}(\xi)$, given by

$$\frac{d\mathbf{r}}{d\xi} = \mathbf{q}, \quad \mathbf{r}(0) = \mathbf{x}_0,$$

where $\mathbf{x}_0$ is the starting point.

Then Equation (2) simplifies to

$$\begin{aligned} \partial_t U + \partial_\xi F(U) &= 0, \\ U(\xi, 0) &= U_0(\mathbf{r}(\xi)), \end{aligned} \tag{4}$$

where $F(U) = [f(u) \quad g(u,v)]$. We denote this as a $triangular$ system since the Jacobian matrix of F is (lower) triangular. The system is solved by front tracking.

where $F(U) = [f(u) \quad g(u,v)]$. We denote this as a $triangular$ system since the Jacobian matrix of F is (lower) triangular. The system is solved by front tracking.

Solution procedure - how to advect the solution one time step:

1. From each grid point generate the streamline backwards ($\xi < 0$). This gives a step function as initial data.
2. Solve the initial value problem up to time $t = \Delta t$ using front tracking, and assign the value at $\xi = 0$ to the grid point.
3. Do some sort of averaging.

Streamlines

There are two approaches for generating streamlines:

- ODE-solvers.
- Analytical methods.

We have used a 4th order Runge-Kutta to generate the streamlines. The method is flexible, but very slow.

Front tracking

Consider a 2×2 system of hyperbolic equations, e.g., Equation (4).

Front tracking is an algorithm for computing a piecewise constant approximation to such systems. A *front* is a jump discontinuity.

Front tracking

Consider a 2×2 system of hyperbolic equations, e.g., Equation (4).

Front tracking is an algorithm for computing a piecewise constant approximation to such systems. A *front* is a jump discontinuity.

Scalar problems: Finite number of steps, even in infinite time.

General systems: Data reduction must be invoked to prevent an infinite buildup of fronts.

The solution consists of a sequences of fronts (shocks) U_{i-1}/U_1 with shock speed σ_i, $i = 1, \ldots, n$.

The algorithm goes as follows:

1. First, u_0 is approximated by a step function.

The algorithm goes as follows:

1. First, u_0 is approximated by a step function.
2. A Riemann problem is associated with each jump in the approximate initial data.

The algorithm goes as follows:

1. First, u_0 is approximated by a step function.
2. A Riemann problem is associated with each jump in the approximate initial data.
3. All Riemann problems are solved. Rarefraction waves are approximated by a series of small fronts. In this way, each Riemann problem produces a sequence of fronts that travel with a finite wave speed.

The algorithm goes as follows:

1. First, u_0 is approximated by a step function.
2. A Riemann problem is associated with each jump in the approximate initial data.
3. All Riemann problems are solved. Rarefraction waves are approximated by a series of small fronts. In this way, each Riemann problem produces a sequence of fronts that travel with a finite wave speed.
4. All fronts are *tracked*. Colliding fronts define new Riemann problems which are solved. Colliding fronts are removed and fronts from the new Riemann problems are added.

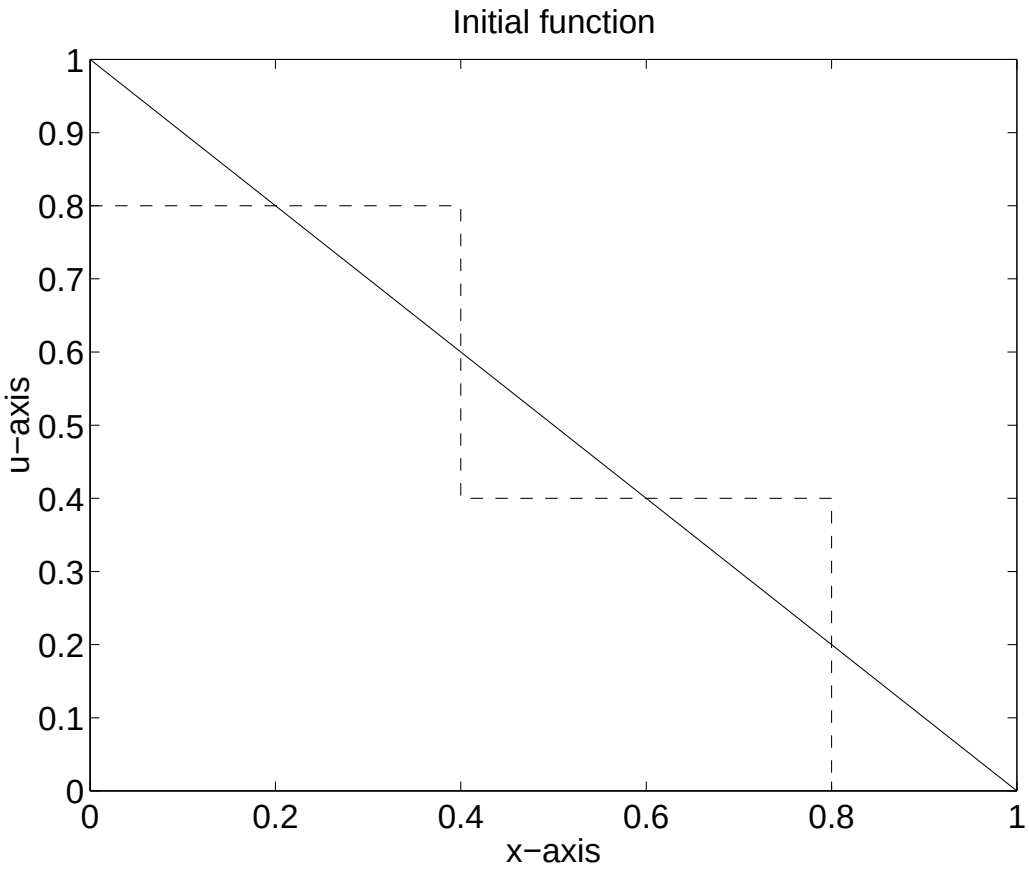
The algorithm goes as follows:

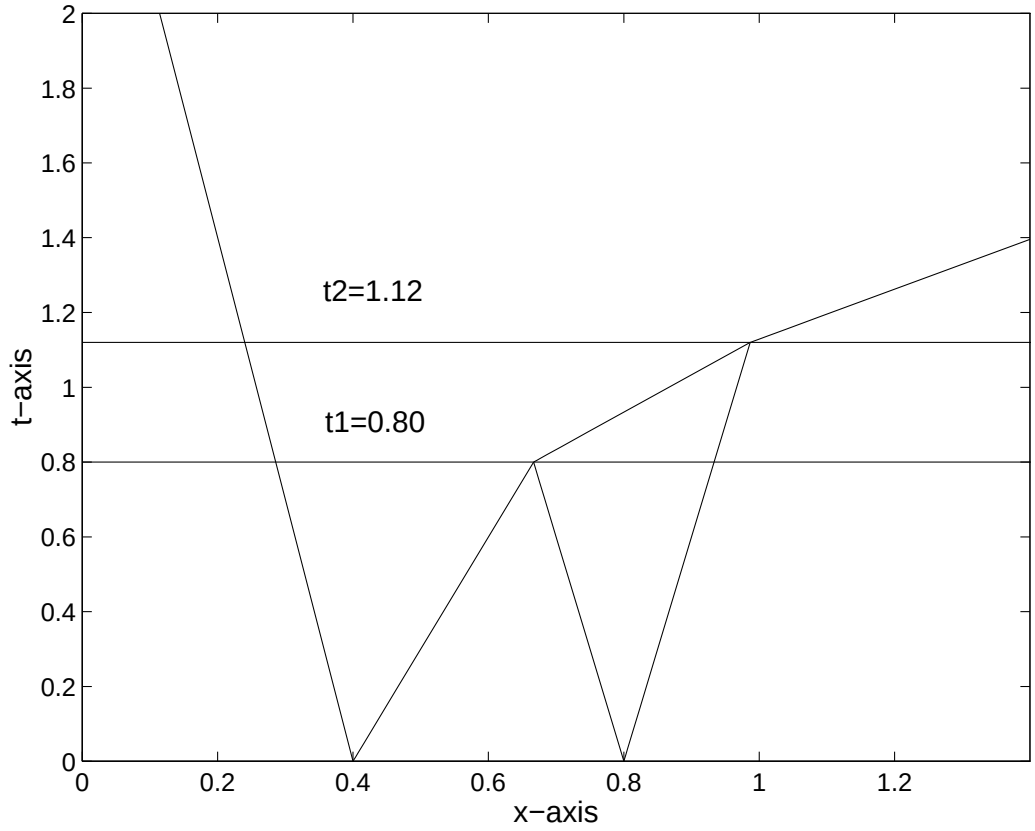
1. First, u_0 is approximated by a step function.
2. A Riemann problem is associated with each jump in the approximate initial data.
3. All Riemann problems are solved. Rarefraction waves are approximated by a series of small fronts. In this way, each Riemann problem produces a sequence of fronts that travel with a finite wave speed.
4. All fronts are *tracked*. Colliding fronts define new Riemann problems which are solved. Colliding fronts are removed and fronts from the new Riemann problems are added.

5. The tracking ends when there are now more collision before final time, or final time is reached.

An illustration:

First the initial data is approximated by a step function. Then the the fronts from the Riemann problems are tracked (next slide).

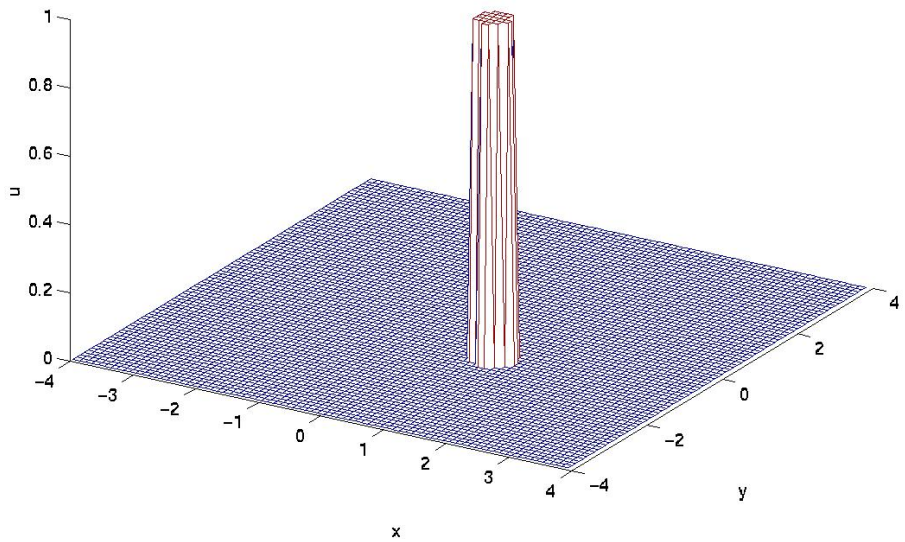



t2=1.12
t1=0.80
t−axis
x−axis
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

EXAMPLE
Rotating sylinder

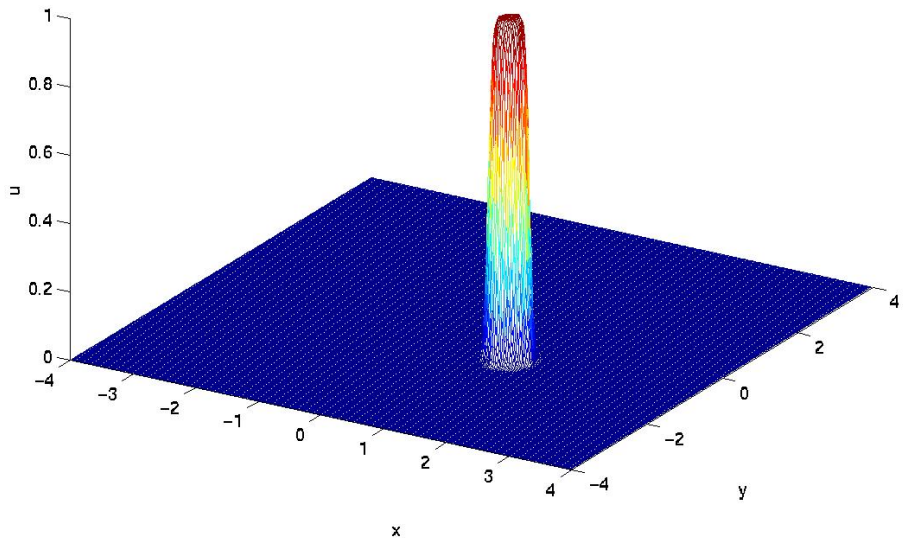
Velocity field:

$$\mathbf{q} = 4\pi[y, -x].$$



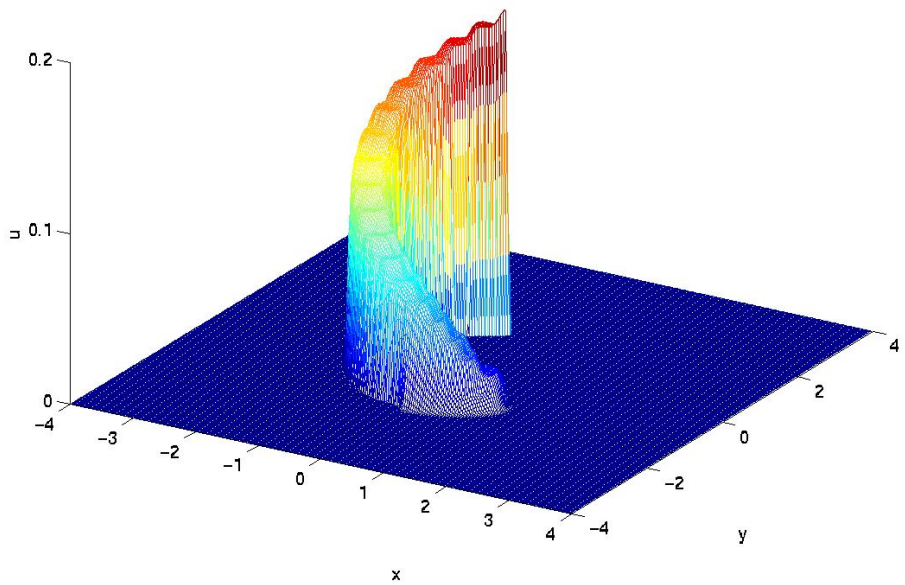
Linear flux:

$$f(u) = 2u$$



Non-linear flux:

$$f(u) = \frac{u^2}{u^2 + (1-u)^2}$$



Diffusion solver

We use a simple second order accurate explicit central-difference scheme since the main objective is to test the tracking concepts.

Note that when using an explicit difference scheme, there is a stability constraint on the time step.

Conclusions

- This is an easy approach to solve (triangular) three phase flow problems in two or more spatial dimensions.

- Things to do:

  - Implement a (semi) analytical method for generating streamlines.
  - What about corrected operator splitting.